A DIGITAL TACHOMETER/ACCELEROMETER

John Hone Bartol, Jr.

United States
Naval Postgraduate School

THESIS

A DIGITAL TACHOMETER/ACCELEROMETER

by

John Hone Bartol, Jr.

December 1969

A Digital Tachometer/Accelerometer

by

John Hone Bartol, Jr.
Lieutenant, United States Navy
B.S., Auburn University, 1962

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
December 1969

ABSTRACT

Analog methods for measuring rotational shaft velocity and acceleration are well developed, yet present many drawbacks in practical use. A device is proposed to digitally compute estimates of the instantaneous velocity and acceleration of a shaft. The method of attack used in arriving at a final design proceeds from development of signal generating equipment through velocity- and acceleration-processing circuitry. Computer simulation is used to optimize the electronic circuitry to achieve the best accuracy.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

I. INTRODUCTION

Devices which accurately measure the instantaneous rotational velocity and acceleration of a shaft have countless engineering applications. One of these is the use of wheel acceleration and velocity measurements as inputs to an aircraft servo braking system whose computer may temporarily override the pilot's braking commands should the plane skid during landing.

Analog methods for measuring shaft speed and acceleration are well developed. A tachometer geared to the shaft being tested develops a dc voltage proportional to the shaft speed while a suitable resistor-capacitor bridge approximately differentiates this voltage to yield an instantaneous acceleration voltage.

The use of analog techniques for shaft measurements has many drawbacks; some of the more objectionable ones are:

1) The number of significant figures in measurement readings is limited to the quality of analog equipment used to generate and display information.

2) Additional equipment (such as a tachometer) added to a shaft increases its inertia.

3) Analog velocity-measurement devices are often nonlinear over the useful range of velocity to be measured.

4) Mechanical wear limits the useful life and reliability of all mechanical/electrical generating units.

Since analog methods offer many problems, a need was seen for the design of a system to digitally measure the instantaneous velocity and acceleration of a shaft. This thesis is the result of the study, design, and simulation of digital electronic equipment to achieve that end.

II. DESIGN PROCEDURE

The design of a digital system to measure shaft acceleration and velocity was in effect the design of a small, special-purpose computer to perform these functions. The design procedure for this study was broken into seven categories:

1) Analysis of design considerations related to rotating shaft measurements; statement of formulae used to express rotary motion and expression of the requirements to be met by the proposed equipment design.

2) Development of an "inertialess" signal generating source whose output is directly proportional to shaft speed.

3) Development of digital circuitry to process the shaft speed signal and produce velocity information.

4) Development of digital circuitry to process the velocity information and produce acceleration information.

5) Computer simulation of the total digital design to yield simulated data input and digital system output information.

6) Analysis of the computer simulation to optimize the system accuracy.

7) Evaluation of the results.

III. DESIGN CONSIDERATIONS

Necessary definitions concerning the rotation of a (rigid) shaft can be easily expressed. In a specific interval of time, a rotating shaft turns through a rotational angle, $\Delta\theta$. The average angular speed, $\bar{\omega}$ is defined

$$\bar{\omega} \equiv \frac{\theta_2-\theta_1}{t_2-t_1} \tag{1}$$

where $\Delta\theta$ is the rotational displacement the shaft has undergone during time Δt. This can also be expressed as

$$\bar{\omega} = \frac{\theta_2-\theta_1}{t_2-t_1} \quad . \tag{2}$$

If Δt approaches 0 as a limit, instantaneous angular speed can be expressed as

$$\omega \equiv \lim_{\Delta t \to 0} \frac{\Delta\theta}{\Delta t} = \frac{d\theta}{dt} \quad . \tag{3}$$

Angular acceleration, α, is the time rate of change of angular velocity. Average acceleration, $\bar{\alpha}$, can be expressed

$$\bar{\alpha} \equiv \frac{\Delta\omega}{\Delta t} = \frac{\omega_2-\omega_1}{t_2-t_1} \tag{4}$$

where ω_2 is final and ω_1 is initial angular speed and t_2 is final time and t_1 is initial time.

When dealing with problems concerning uniformly accelerated rotary motion the following equations apply:

$$\theta = \bar{\omega}t + \theta_1 \tag{5}$$

$$\omega_2 = \omega_1 + \bar{\alpha}t \tag{6}$$

$$\theta_2 = \omega_1 t + \frac{1}{2}\alpha t^2 + \theta_1 \tag{7}$$

where t is elapsed time from initial conditions, and the other variables along with their subscripts are defined above.

The design of a digital device requires that the output be quantized, and that it appear at discrete output times. The design of this special-purpose computer is predicated upon the successful implementation of electronic devices to perform calculations in equations (2) and (4)

$$\bar{\omega} = \frac{\theta_2-\theta_1}{t_2-t_1} \qquad (2)$$

$$\bar{\alpha} = \frac{\omega_2-\omega_1}{t_2-t_1} \quad . \qquad (4)$$

Although it would have been desirable to design a system able to measure any range of rotational speeds, it was felt that design considerations should be predicated on solving a specific problem. The specific problem undertaken was the design of a special-purpose digital computer to measure the rotational speed and acceleration of a magnetic tape transport capstan. The tape transport system concurrently under design by a computer company was to provide an unusual acceleration: 0 through 200 inch/second tape speed over elapsed time of 3.75 msec using a capstan of 0.75 inch radius. To insure that this thesis design might adequately handle this problem, the tape drive requirements were made more demanding.

Tape Drive Requirements:

Drive capstan radius	0.75 inch
Operational tape speed	200 inches/sec
Acceleration time (0-200 inch/sec)	2.5 msec

These data were used to compute average shaft speed and acceleration. The average linear velocity, $\bar{v}$, of the capstan surface can be expressed

$$\bar{v} = r\bar{\omega} \tag{8}$$

where r is the capstan radius and $\bar{\omega}$ is average shaft speed. Converting this equation and substituting values yields

$$\bar{\omega} = \frac{v}{r} = \frac{200 \text{ inch/sec} \times 1 \text{ revolution}}{0.75 \text{ inch} \times 2\pi \text{ radians}} = 42.5 \frac{\text{rev}}{\text{sec}} . \tag{9}$$

The average acceleration can be computed by equation (3).

$$\bar{\alpha} = \frac{\omega_2 - \omega_1}{t_2 - t_1} = \frac{42.5 \text{ rev/sec} - 0 \text{ rev/sec}}{2.5 \times 10^{-3} \text{ sec} - 0 \text{ sec}} = 1.7 \times 10^4 \frac{\text{rev}}{\text{sec}^2} . \tag{10}$$

Allowing for some overdesign it was required that the system be able to measure (at least) the following parameters.

Expected Shaft Speed and Acceleration:

max acceleration - 2×10^4 rev/sec^2

max speed - 100 rev/sec .

IV. DEVELOPMENT OF A SIGNAL GENERATING SOURCE

The many inertialess signal generating methods studied were capable of generating electrical signals directly proportional to shaft position or velocity. A photoelectric design most suitable to the problem solution is hereafter described.

It was anticipated that the capstan drive shaft could have black lines (arc of 0.2° width) easily scribed every 3.6 degrees around its circumference. One small photocell measuring the intensity of reflected light from these lines would produce a resistance inversely proportional to the square root of that intensity. To increase the resolution of this system (from 3.6° to 0.36°) the following scheme was proposed.

As shown in Figs. 1 and 2, there are lines scribed on the capstan above which is positioned a photocell mounting bracket. Nine photocell "windows" (0.2 degrees of arc wide) are cut in this bracket at an equal radius from the capstan center and equally spaced precisely four degrees apart.

Looking at Fig. 2 one can see that only one photocell window has a line completely under it at any time. If the capstan is turned clockwise 0.36 degrees it is easily seen that photocell number two would have directly under it the line which previously had been under photocell number one. The resolution of the system is therefore 0.36°. A shaft rotation of 3.6° would cause ten sequential signals to be developed due to the vernier construction of this device.[1]

---

[1]The photocell envisaged for this system is listed in Industrial Catalog, number 680, p. 83, Allied Electronics, 1968. An "NPN Planar Silicon Light Sensor # LS-400" has the following response times: enable time 1.5 μsec; disable time 15 μsec. Assuming a total cycle time of 20 μsec, the maximum shaft rotational speed with errorless signal generation can be easily calculated for this system.

$$\frac{1}{x \text{ rev/sec} \cdot 100 \text{ lines/rev} \cdot 1 \text{ output signal/line}} \geq 20\ \mu\text{sec} \tag{11}$$

Solving for x yields

$$x \leq \frac{1}{100 \cdot 20 \cdot 10^{-6}} = 500 \text{ rev/sec} \tag{12}$$

This photocell would easily fulfill the requirements of the tape drive system.

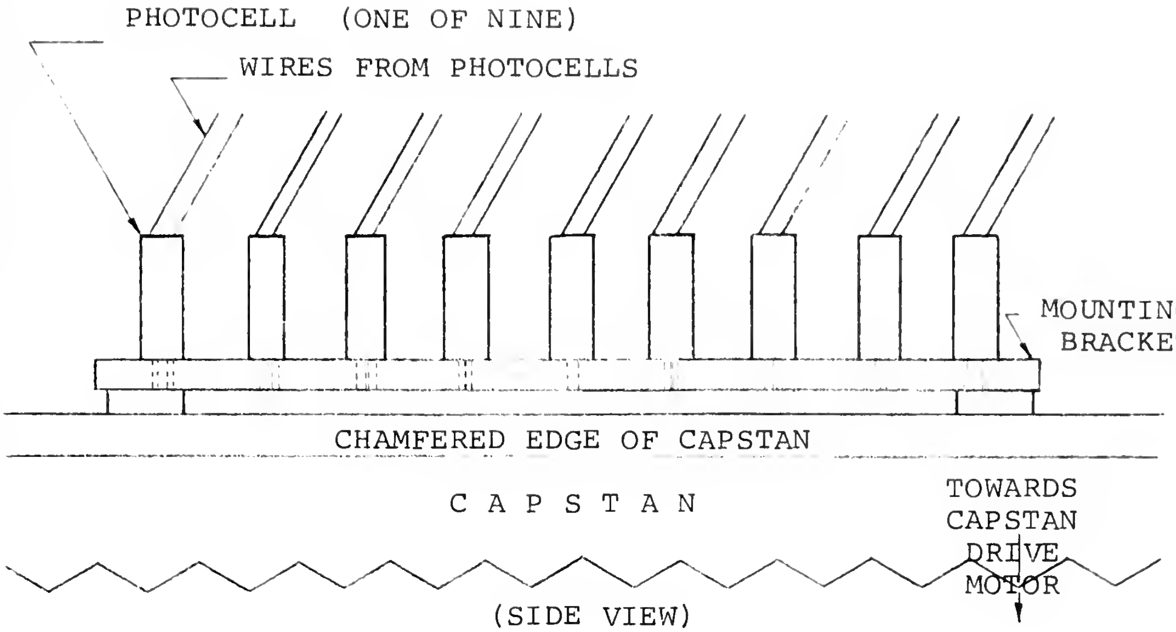


FIG. 1 ENLARGED VIEW OF SIGNAL GENERATOR

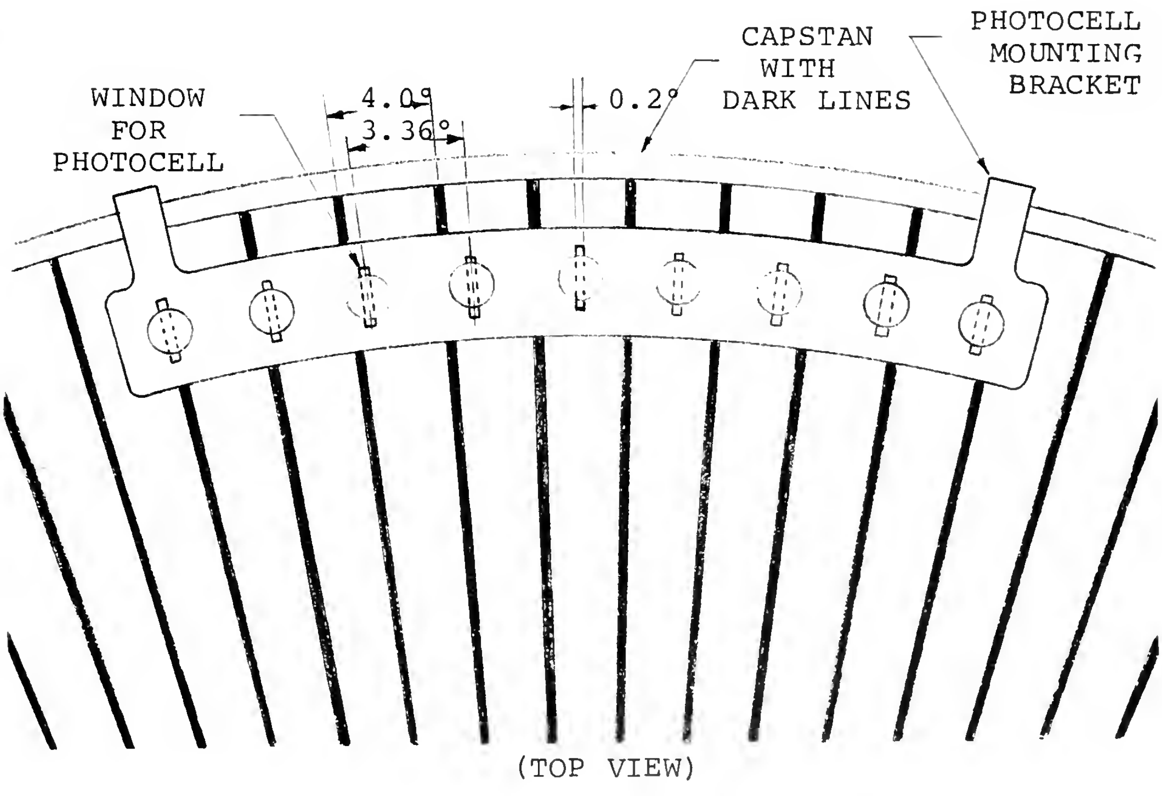


FIG. 2 ENLARGED VIEW OF SIGNAL GENERATOR

The photocell of a "Gossen, Lunasix" photo lightmeter was independently studied and found to vary between 70 ohms and 15 kilohms depending on the amount of light impinging on the cell. Photocells for the system under design could be expected to act similarly. It was anticipated that all nine photocells could be placed in parallel and input to the circuit shown on the following page.

A description of the electronic operation follows. An incremental (0.36 degrees) shaft displacement signal is developed across resistor, R10 and the photocells in parallel. Coupled by C1 to the base of Q1, the positive bias of Q1 is overcome and the transistor is temporarily cut off. The amplified signal is coupled by C2 to a multivibrator composed of the remaining elements. R15 serves as a pulsewidth adjustment, whose time constant is determined by the product of R15 x C3. Diode D2 serves as a voltage stabilizing diode.

In summation, circuitry has been developed to generate a variable pulse width, positive-going square wave that is generated whenever the shaft is rotated every 0.36 degrees.

V. DEVELOPMENT OF CIRCUITRY TO PROCESS THE SHAFT SIGNALS AND OUTPUT VELOCITY INFORMATION.

The development of digital circuitry to process the shaft positional signals involved three important concepts.

1) The shaft signal had to be stored or "buffered" until the processing circuitry had processed it.

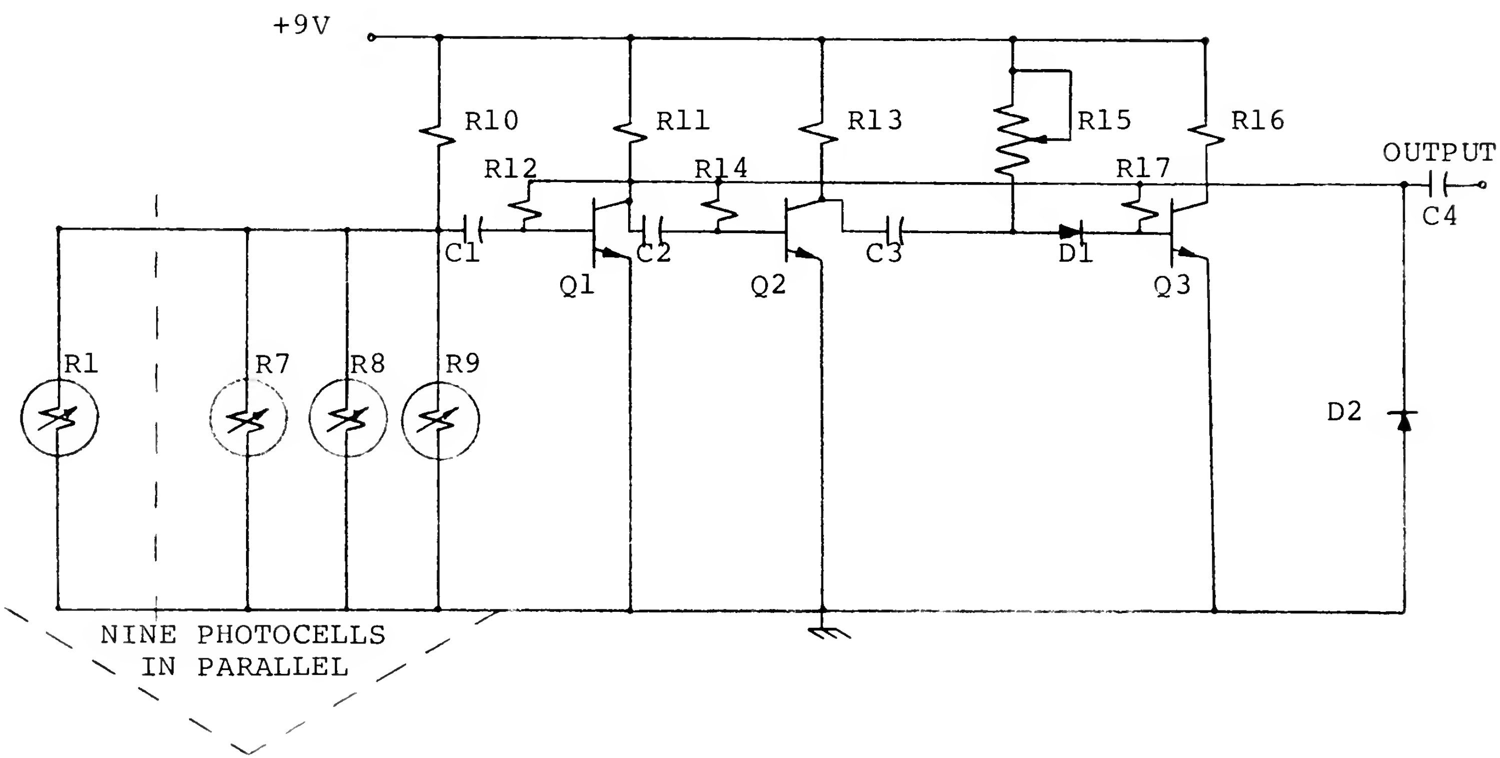


FIG. 3 SHAFT POSITION GENERATOR CIRCUITRY

2) An electronic method of digitally generating the velocity signal had to be developed.

3) After the new velocity signal had been generated, circuitry had to be designed to remove the previous signal stored in the buffer to enable it to receive another output from the signal generator.

The final circuitry anticipated to fulfill the requirements above is shown on the next page.

The digital components proposed for this accelerometer are basic logic elements. The design, theory of operation, and Boolean algebra representation of the basic elements used, such as AND gates, OR gates, JK flip-flops, and shift registers are described in many textbooks [Refs. 1, 3, and 5]. Other equipment such as clock generators, divide-by-500 counters, decoders, and output display equipment are fairly standard. Since the proposed accelerometer was not constructed, a detailed circuit design has not been included.

An explanation regarding individual component functions in relation to system operation can now be made. During the operation of the accelerometer there is received a non-periodic, positive-going square-wave signal for each incremental shaft rotation of 0.36°. If the normal output state for the JK flip-flop is a "0", the occurrence of the square-wave signal at the set, J terminal causes a "1" to occur at the output terminal, Q. (This "1" output remains until the K terminal has a "1" input into it).

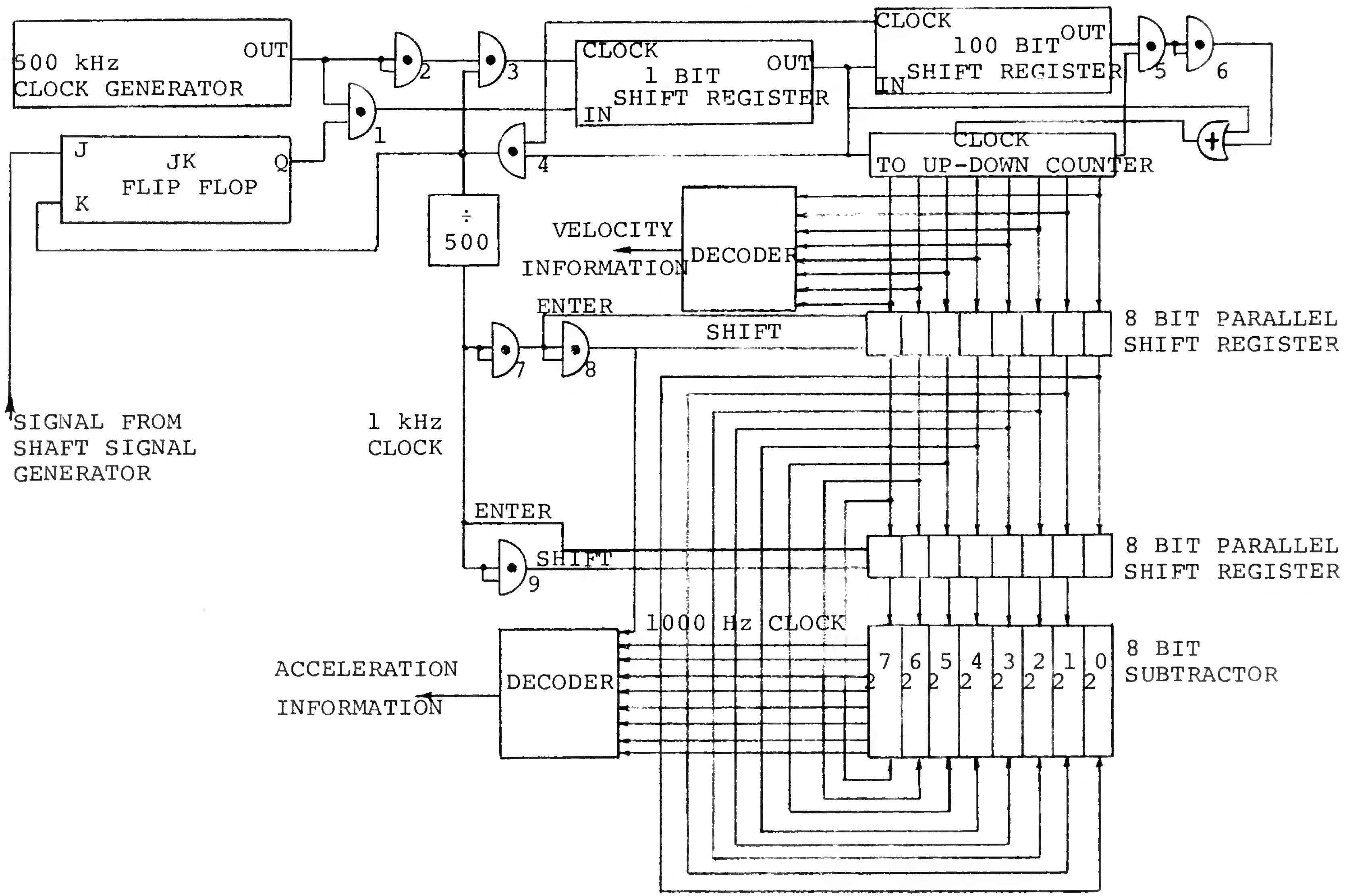


FIG. 4 VELOCITY AND ACCELERATION COMPUTING CIRCUITRY

After the flip-flop output is "1" the occurrence of a positive clock signal causes the output of AND gate number one to be "1". The use of AND gates numbered two and three permits a sufficient time delay to occur such that AND gate number one has reacted long enough to guarantee output of a "1" signal. This delayed clock enables the 1-bit shift register (which already has a "1" input to the J input terminal) and the output shifts from an "0" to a "1". The delayed 500-kHz clock is also fed to AND gate number four. When this delayed clock and the output of the 1-bit shift register are both "1" AND gate number four outputs a "1", which resets the JK flip-flop to the "0" output state. -- An important design requirement should be noted. The output from the shaft signal generator must be less than 1 microsecond in duration (half the clock period) to eliminate the chance of propagating two output signals to the shift register for one input shaft signal. -- Thus circuitry has been designed to buffer the shaft signal until it is processed and sent through the 1-bit shift register, at which time the signal is removed from the flip-flop, preparing it to receive another pulse.

From the circuitry described it can be seen that the 1-bit shift register has 0's or 1's propagated through it at every clock time, a period of two μseconds. From Fig. 5 it can be seen that the 1-bit shift register is followed by a 100-bit shift register. The time for a single "1" to propagate through the long shift register can be calculated.

$$\text{Propagation Time} = \text{number of shift register bits/clock frequency} \quad (13)$$

Thus if a 200-bit shift register were used with a clock frequency of 500 kHz

$$\text{Propagation Time} = 200/5 \times 10^5 = 400\ \mu\text{sec} . \quad (14)$$

The total number of 1's in the shift register is indicative of the average velocity sensed over the propagation time interval. An example best demonstrates this.

Assume the number of shift register bits is 500, and that the clock frequency is 500 kHz. Assume also that the signal sensor has been sensing signals from a shaft which has been continuously rotating at 10 revolutions/second. The following calculations can be made.

$$10 \text{ revolutions/second} \times 360 \text{ degrees/revolution} \times 1 \text{ signal}/0.36 \text{ degrees} = 10{,}000 \text{ signals/second}. \quad (15)$$

$$\text{Propagation Time} = \frac{500 \text{ shift register bits}}{5 \times 10^5 \text{ cycles/sec} \times 1 \text{ shift/cycle}} = 1000\ \mu\text{sec}. \quad (16)$$

$$10^4 \text{ signals/second} \times 10^{-3} \text{ sec} = 10 \quad \text{1's stored in the shift register} . \quad (17)$$

From this example, the number of 1's stored in the long shift register directly corresponds to the shaft velocity being measured. The method of adding the 1's in the shift register is effectively handled by the use of an "up-down" counter, a device which has the capability of increasing or decreasing its count when the appropriate "up" or "down" command is present. The occurrence of the delayed 500 kHz clock causes the serial shift register to simultaneously shift right by one bit position all of its stored bits. AND gates five and six are included to prevent "up" and "down" signals arriving simultaneously, a condition not permitted in up-down counters.

Circuitry developed to this point has stored in the up-down counter the total number of electronic signals corresponding to an average shaft velocity. These signals are sensed, decoded (to decimal numbers), and displayed at every clock time. Thus circuitry has been designed to process the shaft speed signal and display velocity information.

VI. DEVELOPMENT OF CIRCUITRY TO PROCESS ACCELERATION INFORMATION

From equation (4) it can be seen that average acceleration over an interval is the change in angular shaft velocity divided by the time interval. The development of digital circuitry to perform this calculation involved three important concepts.

1) An electronic method of generating the acceleration output period had to be developed.

2) The shaft velocity had to be sampled and stored once each period.

3) An electronic method for calculating and displaying the acceleration value once each period had to be devised.

Fig. 4 includes the circuitry to perform these functions. Individual component operation can now be discussed.

Assume a shaft velocity has been calculated and is therefore stored in the up-down counter. Assume also that at time "t1" a velocity has been stored in both 8-bit shift registers. The divide-by-500 counter effectively generates a 1000-Hz clock frequency that is synchronized with the master (500kHz) clock. When the next 1000 Hz clock pulse is generated at time "t2", this velocity is shifted (after AND gate number nine's time delay) into the positive terminal of the subtractor. After AND gate number seven's time delay the new velocity in the up-down counter is sampled and stored in the 8-bit shift register. After AND gate number eight's time delay this velocity is shifted simultaneously to the output terminals of the 8-bit shift register one as well as to the minus terminals of the subtractor. After AND gate number 10's time delay the subtractor is enabled, permitting the subtraction of velocities sampled at times "t2" and "t1". The input to the acceleration decoder is therefore the difference in velocities sampled at millisecond intervals. The value 1000 as a scale factor must be used to yield true acceleration.

Circuitry has been devised to calculate average shaft acceleration at 1-millisecond intervals. Although information displayed at this rate is too fast for human observation, it is anticipated that this acceleration information would be electronically incorporated with a servo control system.

VII. COMPUTER SIMULATION OF THE SYSTEM

It was anticipated that the system's calculations of velocity and acceleration would have some error due to the quantizing necessary in a digital system. It was expected that these errors would have some complicated relationship to the number of shift register bits used to store the 1's and 0's corresponding to shaft velocity. A computer simulation was written to experimentally determine this relationship.

A printout of the simulation used is printed under the Computer Simulation Program Listing section. The most important simulation variable names are listed below with a description of their function.

Variable	Purpose	Units
TIME	Serves as the independent time base for the simulation.	seconds
DELT	Serves as the incremental time base for the simulation.	seconds
FINTIM	Serves as the final time to be simulated. The total real time simulation was 5.15 milliseconds.	seconds
CKOUT	Serves as the master 500kHz clock; it is a square wave of amplitude 1.0 and has a period of two microseconds.	
THETA	Serves as the true angular position of the simulated shaft which is uniformly accelerated from rest in accordance with equation (7).	degrees
OMEGA	Serves as the instantaneous true shaft velocity.	degrees/second

Variable	Purpose	Units
OMEGA0	Serves as the initial shaft velocity at time zero.	degrees/second
ALFA	Serves as the true simulated acceleration.	degrees/second^2
ALRPS2	Serves as the true simulated acceleration.	$\text{revolutions/second}^2$
RPST	Serves as the true velocity of the simulated shaft.	revolutions/second
SPIKE	Serves as the electrical output voltage developed across the nine photocells in the Signal Generator. -There is a one-volt "spike" developed when the shaft has turned each 0.36 degrees.	volts
SHIN	Serves as the electrical voltage output as a 0.9-millisecond positive-going square wave from the Shaft Position Generator Circuitry shown in Fog. 3.	volts
FFOUT	Serves as the binary output from the JK flip-flop. (FFOUT is either 0 or 1.)	
IDD2	Serves as the delay time required for the JK flip-flop to change its output from 0 to 1. (IDD2 expresses in integer form, the number of DELT simulation times contained in an average JK flip-flop enable time.)	
IDD3	Serves as the delay time required for a JK flip-flop to change its output from 1 to 0. (IDD3 expressed in integer form, the number of DELT simulation times contained in an average JK flip-flop disable time.)	
BFOUT	Serves as the output from AND gate number 1.	
IDD4	Expresses the integer number of DELT simulation times contained in an average AND gate enable time.	

Variable	Purpose	Units
IDD5	Expresses the integer number of DELT simulation times contained in an average AND gate disable time.	
DCK	Serves as the delayed 500kHz clock after going through AND gates two and three.	
SR1OUT	Serves as the output from the 1-bit shift register.	
SHRSET	Serves as the output from AND gate number four; its effect is to reset the JK flip-flop to output 0 and enable it to receive another SHIN signal.	
NUM	Serves as the integer number of shift register bits being simulated.	
MINUS	Serves as the output from the 100-bit shift register. When MINUS is 1, the total stored in the up-down counter is decreased by a value of 1.	
RPSE	Serves as the shaft velocity as stored in the up-down counter.	revolutions/second
STORE1	Serves as the velocity stored in the 8-bit shift register number 1.	revolutions/second
STORE2	Serves as the velocity stored in the 8-bit shift register number 2.	revolutions/second
ACCEL	Serves as the output from the acceleration decoder.	revolutions/second
MAINN	A subroutine which serves as the main program. The main program (which calls MAINN) serves as a dummy program whose purpose is to dimension a matrix which corresponds to the 100-bit shift register.	

Variable	Purpose	Units
STPLOT	A subroutine which graphically displays system responses. (The sample time used for this simulation is 1 sample/20 microseconds to permit photographic reduction of the results for inclusion in the thesis.)	

VI. EVALUATION OF THE RESULTS

Fifteen computer calculations were made to simulate the system output while using the following number of shift register bits: 32, 40, 64, 84, 100, 108, 120, 130, 150, 200, 250, 350, 400, 500, and 640. Average velocity dispersion data were derived by drawing two straight lines through the most dispersed "RPSE" points on the computer graphs, Fig. 7. The vertical distance between these lines was averaged (for 1, 2, 3, 4, and 5 millisecond times) and divided by two to yield the average maximum distance from the mean. Two straight lines were drawn through the top and bottom points of the character used to plot the curve "RPST". The vertical distance between these two lines was subtracted from the average maximum distance derived above. This remaining distance, when multiplied by the proper graph scale, 10.0 units/inch, yielded the average velocity dispersion. Two lines were plotted midway between each pair of dispersion lines mentioned above yielding accurate graphical depictions of true and estimated velocity. The vertical distance between these two lines was averaged and scaled to yield the average velocity error. Average acceleration data were derived from averaging the written computer acceleration output values computed after one millisecond. The averaged error data collected from this analysis are listed below.

Number of Shift Register Bits	Average Velocity Error (Rev/Sec)	Average Velocity Dispersion (Rev/Sec)	Average Acceleration (Rev/Sec^2)
32	- 3.300	13.85	1.875×10^4
40	- 3.180	11.35	1.750
64	- 2.862	6.26	1.875
84	- 2.940	5.31	1.905
100	- 2.531	3.72	1.900
108	- 3.158	3.58	1.945
120	- 3.280	3.17	1.917
130	- 3.412	2.86	1.923
150	- 3.593	2.30	1.933
200	- 4.891	1.90	1.960
250	- 4.500	1.48	1.880
350	- 6.844	1.08	1.857
400	- 8.688	0.91	1.850
500	-10.063	0.53	1.800
640	-11.187	0.52	1.750

These data are depicted graphically in Fig. 5.

An evaluation of the computer results can now be made. Fig. 5 shows that the average acceleration peaks close to an average of 1.9×10^4 revolutions/second^2 (versus 2×10^4 which was simulated) for a shift register bit length of 100. The average velocity error reaches a minimum and has a velocity dispersion of less than four revolutions/second for this same number of shift-register bits. For these reasons, it is recommended that the final shift-register bit length be made one hundred.

Two amplifying points should be made to stress this determination. The high dispersion caused by simulation of a 40-bit register caused a large acceleration error. An error such

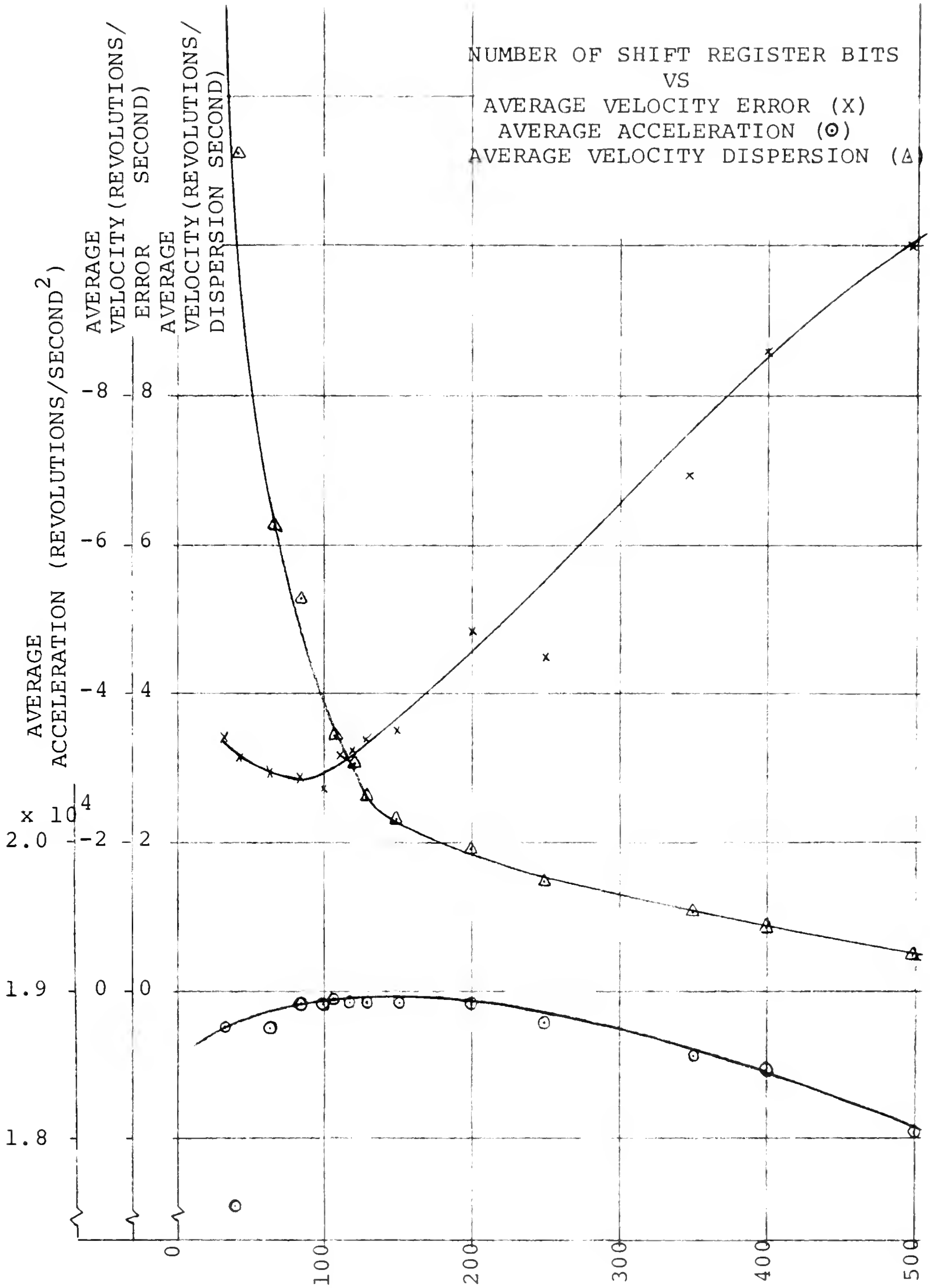


FIG. 5 RESULTS OF COMPUTER ANALYSIS

as this would stress the point of using a low number of shift-register bits. The choice of using 100 bits, on the other hand, appears to yield a better calculation of acceleration. Although not shown in Fig. 5, the acceleration calculated by the system for a 100-bit shift register is exactly 2.0×10^4 for the last 4 milliseconds.

As a final evaluation of the system a frequency response of the velocity computing circuitry was simulated on the computer.[2] A summary of the changes made is listed below.

1) Initial shaft velocity was 42.5 revolutions/second.

2) The shaft velocity was sinusoidally modulated about this average shaft speed at frequencies of interest.

3) The instantaneous accelerations needed to achieve these shaft velocity changes were calculated and plotted

---

[2]It was unfeasible to make a frequency response simulation of the acceleration computing circuitry due to two factors. Since the acceleration calculations were made for every millisecond of simulated time, a significant number of individual acceleration calculations was limited to the amount of real computer time available for an extended study. The amount of significant experimental evidence to be gained from at least 70 minutes of anticipated Central Processing Unit time on the IBM 360/67 was considered too expensive to justify the study. More important, however, is the fact that the acceleration sampling rate imposes a limit on the variations that can be measured accurately. Specifically, C. E. Shannon's Sampling Theorem states that "the sampling frequency must be at least twice as great as the frequency of the highest Fourier component of the analog signal being sampled." [Ref. 2] In other words, this accelerometer cannot be expected to calculate accurately accelerations which cause a shaft to change its response at a rate exceeding 500 Hz.

4) The input shaft velocities and the system responses were simultaneously plotted for later frequency response calculations.

The actual computer simulation changes used are listed below.

Change		Computer Format		Card Number
1.		DATA TIME, OMEGA0,START/0., 15300., 0./	*	00027
2.		FREQMD = (the modulation frequency simulated)	*	00066
3.		OMEGA1 = OMEGA	*	0227A
4.		OMEGA = OMEGA0 * COS (TWOPI*FREQMD*TIME) + OMEGA0	*	00233
5.		ALFA = (OMEGA - OMEGA1) / DELT	*	0233A
6.		2 T79, 'RPST', T88, 'ALFA' /)	*	00388
7.	4000	FORMAT(1X, I6, 1PE13.6, T34, 1PE13.6, T55,	*	00394
8.		* 1PE13.6, T79, 1PE13.6, 2X, 1PE13.6)	*	0394A

Ten computer simulations were made to determine the response for the velocity computing circuitry. The results are tabulated below and graphed on the next page.

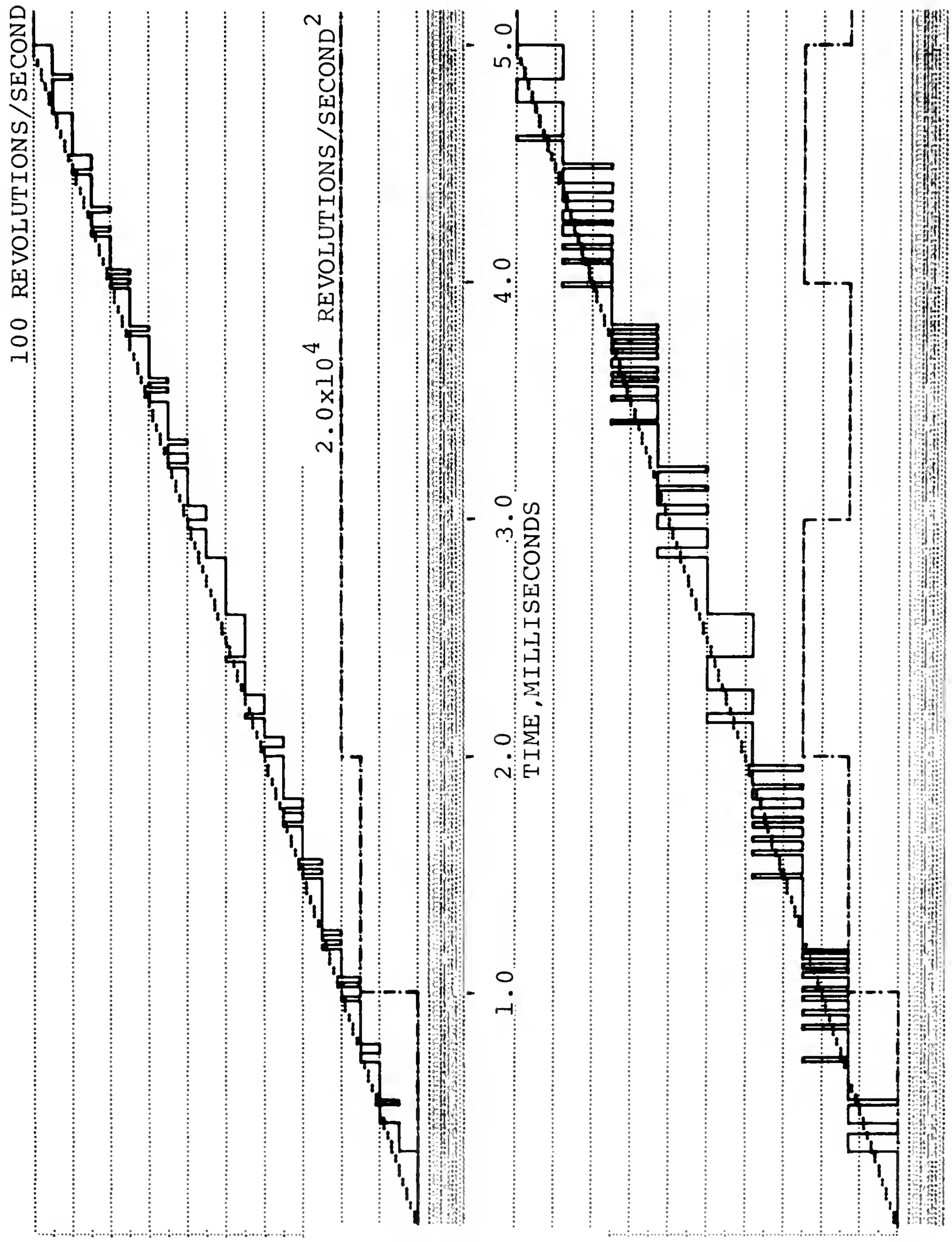
Modulation Frequency (Hz)	Response (decibels)	Phase Lag (degrees)
20	0.	undeterminable
60	0.	less than 2.0
150	0.	6.42
500	0.	17.9
850	0.	32.2
1300	- 0.54	43.5
1800	- 0.54	63.5
2000	- 1.17	66.9
2500	- 1.89	90.0
3500	- 3.85	119.8
5000	-12.24	undeterminable

Fig. 7 demonstrates well that the system has a usable output far in excess of the 500 Hz. restriction imposed by the acceleration sampling rate. It should be noted that the shaft acceleration necessary to produce a 500 Hz. modulation is 1.334×10^5 revolutions/second^2. Since Fig. 7 shows that the proposed system can perform well for modulation frequencies of 500 Hz., the anticipated use of velocity and acceleration calculations which vary at much slower rates is expected to be reliable.

RPST ······· RPSE — ACCEL -·-·-

FIG.6 REAL TIME DISPLAY OF SYSTEM OUTPUT

NUM=100 NUM=40



COMPUTER OUTPUT

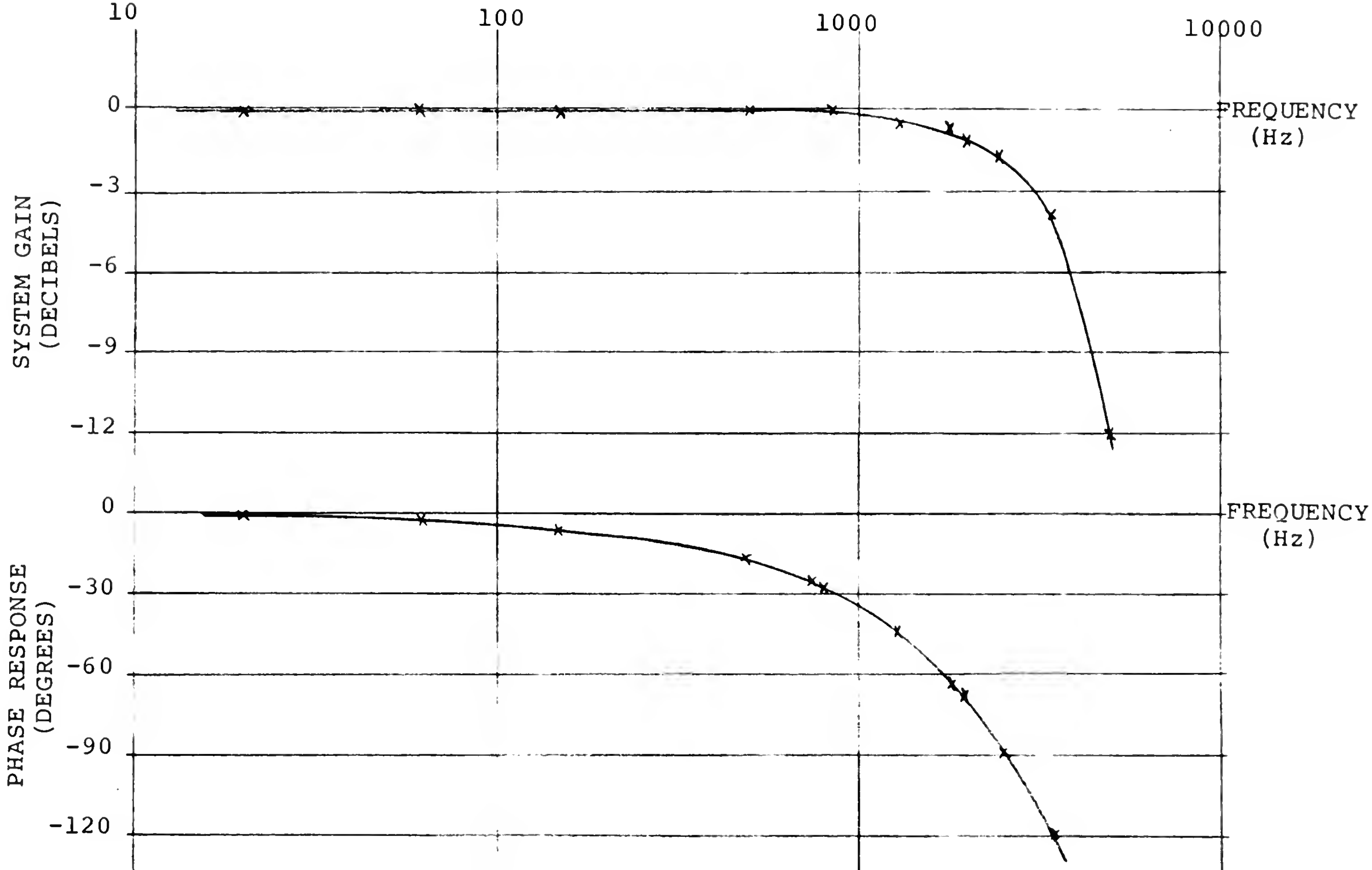


FIG. 7 FREQUENCY RESPONSE OF OPTIMIZED VELOCITY COMPUTING CIRCUITRY

COMPUTER PROGRAM LISTING

```
//BOX11#T   JOB (0230,0390FT,CDP7),'BARTOL,J.H.',CLASS=K                        *
// EXEC FORTCLG,PARM.FORT='MAP       ',REGION.GO=100K,TIME.GO=10               *
//FORT.SYSIN DD *                                                               *
C                                                                               *
C                                                                               *
CP67USERID  0230811      10/30/69     16.16.23      *************************  *
      DIMENSION SR(640),P(2000,4)                                               *
      REAL*4 NUM                                                                *
      READ(5,4999)NUM                                                           *
 4999 FORMAT(F15.0)                                                             *
      WRITE(6,4998)NUM                                                          *
 4998 FORMAT(/1X,'NUM= ',F15.0/)                                                *
      INUM=NUM                                                                  *
      NNN=2000                                                                  *
      CALL MAINN(INUM,SR,NNN,P)                                                 *
      STOP                                                                      *
      END                                                                       *


      SUBROUTINE MAINN(INUM,SR,NNN,P)                                           *
      COMMON MAX(1,10),MXTIME(1,10),MIN(1,10),MNTIME(1,10)                      *
      DIMENSION A(100),D( 100),E(2000),DMINUS(2),P(NNN,4)                       *
      DIMENSION SR(INUM),AV(113,2),AVV(113)                                     *
      REAL*4 NUM,MINUS,MTIME,MAX,MIN,MXTIME,MNTIME,NTIME                        *
      INTEGER*2 TITLE(60)                                                       *
      INTEGER*2 TITLE1(60)                                                      *
      DATA RPSE/0./,RPST/0./,FFOUT/0./,SHRSET/0./,GTIME/1./                     *
      DATA SHIN,THETA0,CKOUT,BFOUT,SR1OUT/0.,0.,0.,0.,0./                       *
      DATA TIME,OMEGA0,START/0.,0.,0./                                          *
      DATA AND/0./,DCK/0./,SR1OUT,UPCCUN/0.,0./                                 *
      DATA JA,ATIME/1,0./,JP/10/                                                *
      DATA LNCNT,PTIME/0,1.1/                                                   *
      DATA LNSKIP/0/                                                            *
      DATA PLNCNT/0./                                                           *
      DATA MINUS/0./                                                            *
      DATA ADD/0./                                                              *
      DATA ISHNST/0/                                                            *
      DATA RMINU/0./                                                            *
      DATA ACCEL,        STORE1/0.,0.    /                                      *
      DATA RJST/1./,PRJST/1.E-03/                                               *
      DATA JG/10/                                                               *
      ZERO=0.                                                                   *
      DO 8011 I=1,113                                                           *
      AV(I,1)=0.                                                                *
      AV(I,2)=0.                                                                *
 8011 AVV(I)=0.                                                                 *
      DO 8009 I=1,4                                                             *
      DO 8010 J=1,2000                                                          *
```

```
 8010 P(J,I)=0.
 8009 CONTINUE
      DO 8008 I=1,100
 8008 A(I)=0.
      DO 8007 I=1,100
 8007 D(I)=0.
      DO 8006 I=1,2000
 8006 E(I)=0.
      DO 8005 J=1,INUM
      SR(J)=0.
 8005 CONTINUE
      DO 8004 K=1,10
      MAX(1,K)=-10.E60
      MXTIME(1,K)=0.
      MIN(1,K)=+10.E60
      MNTIME(1,K)=0.
 8004 CONTINUE
      DELT=5.E-06
      FINTIM=3.0E-04
      ALFA=7.2E06
C     ALFA IS IN (DEGREES/SEC**2)
C     ALRPS2 IS IN (REV/SEC**2)
      ALRPS2=ALFA/360.
      SHINPW=.9E-06
      TFFUP=3.E-08
      TCK=7.E-08
      TSR1=3.E-08
      TFFDN=3.5E-08
      TANDUP=3.0E-08
      TANDDN=2.5E-08
      TSR1UP=3.0E-08
      TSR1DN=4.E-08
      TDELAY=7.0E-08
      READ(5,8003)CLKFRQ
 8003 FORMAT(F10.0)
      WRITE(6,8002)CLKFRQ
 8002 FORMAT(1X,'CLKFRQ=',F10.0)
      TWOPI=6.2831853
      T=1./CLKFRQ
      R111=1./111.
      NUM=INUM
      TLP=T*NUM
      RMULT=500./NUM
      IINUM=NUM+1.
 1100 CONTINUE
      D1=SHINPW/DELT
      ID1=D1
      IF(ID1) 985,985,984
```

```
985  WRITE(6,983)
983  FORMAT(T2,'SHINPW .LE. DELT')
984  DD1=ID1
     IF(D1.GT.DD1) ID1=ID1+1
     D2=TFFUP/DELT
     ID2=D2
     IF(ID2) 982,982,981
982  WRITE(6,980)
980  FORMAT(T2,' TFFUP .LE. DELT')
981  DD2=ID2
     IF(D2.GT.DD2) ID2=ID2+1
     IF(ID2.GE.100) GOTO 978
     GOTO 977
978  WRITE(6,976)
976  FORMAT(T2,' TFFUP IS TOO BIG')
     GOTO 1000
977  CONTINUE
     D3=TFFDN/DELT
     ID3=D3
     IF(ID3) 975,975,974
975  WRITE(6,973)
973  FORMAT(T2,' TFFDN .LE. DELT')
974  DD3=ID3
     IF(D3.GT.DD3) ID3=ID3+1
     IF(ID3.GE.100) GOTO 968
     GOTO 967
968  WRITE(6,966)
966  FORMAT(T2,' TFFDN IS TOO BIG')
     GOTO 1000
967  CONTINUE
     D4=TANDUP/DELT
     ID4=D4
     IF(ID4) 959,959,958
959  WRITE(6,957)
957  FORMAT(T2,' TANDUP .LE. DELT')
958  DD4=ID4
     IF(D4.GT.DD4) ID4=ID4+1
     IF(ID4.GE.100) GOTO 955
     GOTO 954
955  WRITE(6,953)
953  FORMAT(T2,' TANDUP IS TOO BIG')
     GOTO 1000
954  CONTINUE
     D5=TANDDN/DELT
     ID5=D5
     IF(ID5) 971,971,970
971  WRITE(6,969)
969  FORMAT(T2,' TANDDN .LE. DELT')
```

```
970 DD5=ID5
    IF(D5.GT.DD5) ID5=ID5+1
    IF(ID5.GE.100) GOTO 964
    GOTO 963
964 WRITE(6,962)
962 FORMAT(T2,' TANDON IS TOO BIG')
    GOTO 1000
963 CONTINUE
    D7=TDELAY/DELT
    ID7=D7
    IF(ID7) 950,950,949
950 WRITE(6,948)
948 FORMAT(T2,' TDELAY .LE. DELT')
949 DD7=ID7
    IF(D7.GT.DD7) ID7=ID7+1
    IF(ID7.GT.100) GOTO 947
    GOTO 946
947 WRITE(6,945)
945 FORMAT(T2,' TDELAY IS TOO BIG')
    GOTO 1000
946 CONTINUE
    D8=TSR1UP/DELT
    ID8=D8
    IF(ID7) 940,940,939
940 WRITE(6,938)
938 FORMAT(T2,' TSR1UP .LE. DELT')
939 DD8=ID8
    IF(D8.GT.DD8) ID8=ID8+1
    IF(ID8.GT.100) GOTO 937
    GOTO 936
937 WRITE(6,935)
935 FORMAT(T2,' TSR1UP IS TOO BIG')
    GOTO 1000
936 CONTINUE
    D6=T/DELT
    ID6=D6
    IF(ID6) 920,920,919
920 WRITE(6,918)
918 FORMAT(T2,' T .LE. DELT')
919 DD6=ID6
    IF(D6.GT.DD6) ID6=ID6+1
    D9=TSR1DN/DELT
    ID9=D9
    IF(ID9) 910,910,909
910 WRITE(6,908)
908 FORMAT(T2,' TSR1DN .LE. DELT'/)
909 DD9=ID9
    IF(D9.GT.DD9) ID9=ID9+1
```

```
      SRDEL=T*(NUM-1.)                                                  *
      D10=SRDEL/DELT                                                    *
      ID10=D10                                                          *
      DD10=ID10                                                         *
      IF(D10.GT.DD10) ID10=ID10+1                                       *
      WRITE(6,723) ID1,ID2,ID3,ID4,ID5                                  *
  723 FORMAT(5I12)                                                      *
      WRITE(6,722) ID6,ID7,ID8,ID9,ID10                                 *
  722 FORMAT(5I12///)                                                   *
      IF(START.EQ.1.) GOTO 6000                                         *
      OMEGA3=TWOPI*CLKFRQ                                               *
      IDD2=100-ID2                                                      *
      IDD3=100-ID3                                                      *
      IDD4=200-ID4                                                      *
      IDD5=300-ID5                                                      *
      IDD6=400-ID5                                                      *
      IDD7=500-ID7                                                      *
      IDDD8=800-ID8                                                     *
      IDD8=600-ID8                                                      *
      IDD10=700-ID4                                                     *
      ID69=ID6+ID9                                                      *
C     ******************************************************************
      DO 998 I=1,1000000                                                *
C     IF(TIME.GT.2.5E-03) ALFA=0.                                       *
C     IF(TIME.GT.3.69E-03) ALFA=-5.0E06                                 *
  999 FCLOCK=OMEGA3*TIME                                                *
      RI=I                                                              *
      OSC=SIN(FCLOCK)                                                   *
      IF(OSC.GT.0.) GOTO 1                                              *
      IF(OSC.LE.0.) GOTO 2                                              *
    1 SQUARE=1.                                                         *
      GOTO 3                                                            *
    2 SQUARE=0.                                                         *
    3 CONTINUE                                                          *
      CKOUT=SQUARE                                                      *
      IF(TIME.EQ.0.)OMEGA=OMEGAO                                        *
      IF(TIME.EQ.0.) THETA=THETAO                                       *
      THETA=OMEGA*DELT+.5*ALFA*DELT**2+THETA                            *
      IF(OMEGA.EQ.0.)GOTO 15                                            *
      RPST=OMEGA/360.                                                   *
   15 CONTINUE                                                          *
      IF(TIME.EQ.0.)OMEGA=OMEGAO                                        *
      OMEGA=OMEGA+ALFA*DELT                                             *
      IF(TIME.EQ.0.)OMEGA=OMEGAO                                        *
      IPUL=AINT(THETA /.36)                                             *
      N1=IPUL                                                           *
      A(2)=A(1)                                                         *
      A(1)=N1                                                           *
```

```
      DELSHN=A(1)-A(2)
      IF(DELSHN.GE.1.) GOTO 993
      SPIKE = 0.
      GOTO 992
  993 SPIKE=1.
  992 IF(DELSHN) 991,991,4
  991 SPIKE=0.
      MISPUL = 0.
      GOTO 6
    4 CONTINUE
      IF (DELSHN .GT. 1.) GOTO 5
      MISPUL = 0.
      GOTO 6
    5 MISPUL = DELSHN    - 1.
    6 CONTINUE
      MISPUL=0.
      IF(SPIKE.EQ.0.) GOTO 990
      IF(SPIKE.GT..9) ISHNST=I+ID1
  990 SHIN=0.
      IF(I.LT.ISHNST) SHIN=1.
      DO 979 K=1,99
  979 D(K)=D(K+1)
      D(100)=SHIN
      FFUP=D(IDD2)
      DO 972 K=1,99
  972 E(K)=E(K+1)
      E(100)=SHRSET
      FFDN=E(IDD3)
      IF(FFUP.EQ.0..AND.FFDN.EQ.0.) GOTO 11
      IF(FFUP.EQ.0..AND.FFDN.EQ.1.) GOTO 8
      IF(FFUP.EQ.1..AND.FFDN.EQ.0.) GOTO 9
      IF(FFUP.EQ.1..AND.FFDN.EQ.1.) GOTO 10
      GOTO 11
    8 FFOUT=0.
      GOTO 11
    9 FFOUT=1.
      GOTO 11
   10 MISPUL=1.
   11 CONTINUE
      IF(FFOUT.EQ.1..AND.CKOUT.EQ.1.) GOTO 12
      IF(BFOUT.EQ.1.) GOTO 13
      BFOUT=0.
      BFUP=0.
      BFDN=1.
      GOTO 14
   12 BFUP=1.
      DO 960 K=101,200
  960 E(K )=E(K+1 )
```

```
      E(200)=BFUP
      BFUP1=E(IDD4)
      BFOUT=BFUP1
      BFDN=1.
      GOTO 14
   13 BFDN=0.
      DO 965 K=201,300
      E(K )=E(K+ 1)
  965 E(300)=BFDN
      BFDN1=E(IDD5)
      BFOUT=BFDN1
      BFUP=0.
   14 CONTINUE
      BFOUT1=BFOUT
      DO 961 K=301,400
  961 E(K )=E(K+ 1)
      E(400)=CKOUT
      CKD=E( IDD6)
      IF(CKD.EQ.0..AND.BFOUT1.EQ.1.) BFOUT=0.
      DO 944 K=401,500
  944 E(K)=E(K+1)
      E(500)=CKOUT
      DCK=E( IDD7)
      SP1=DCK
      A(3)=A(4)
      A(4)=SP1
      SP2=A(3)
      ENABLE=SP1-SP2
      SR1UP=0.
      IF(ENABLE.EQ.1..AND.BFOUT.EQ.1.) SR1UP=1.
      DO 851 K=701,800
  851 E(K)=E(K+1)
      E(800)=ENABLE
      DENABL=E(800-ID8)
      DO 850 K=501,600
  850 E(K)=E(K+1)
      E(600)=SR1UP
      SR1U=E( IDD8)
      IF(SR1U) 930,930,929
  929 IPP=I+ID69
      SR1OUT=1.
  930 CONTINUE
      IF(I.GT.IPP) SR1OUT=0.
      AND=0.
      IF(SR1OUT.EQ.1..AND.DCK.EQ.1.) AND=1.
      DO 890 K=601,700
  890 E(K)=E(K+1)
      E(700)=AND
```

```
      SHRSET=E(IDD10)
      ADD=0.
      IF(SR1OUT.GT..9) ADD=1.
      A(6)=A(5)
      A(5)=ADD
      DADD=A(6)
      IF(SR1OUT.GT..9.AND.DADD.GT..9) ADD=0.
      UPCOUN=UPCOUN+ADD
      IF(SR(2).GT..9)SR(INUM)=0.
      IF(ADD.GT..9)SR(INUM)=1.
      IF(DENABL.GT..995)GOTO 889
      GOTO 892
  889 DO 891 J=1,INUM
  891 SR(J)=SR(J+1)
      RMINU=-SR(1)
      IRMINU=RMINU
      IMINUS=MINUS
      MINUS=IRMINU+IMINUS
  892 RPSE=(UPCOUN+MINUS)*RMULT
      DELRPS=RPSE-RPST
      IF(I.GT.8500.AND.I.LT.8550)PTIME=1.1
      JATIME=TIME*1.E3
      IF(JATIME.GE.JA )GOTO 886
      GOTO 885
  886 JA=JA+1
      STORE2=STORE1
      STORE1=RPSE
      ACCEL=(STORE1-STORE2)*1.E3
  885 CONTINUE
      IF(TIME.LE.(FINTIM+DELT))GOTO 6000
      READ(5,5004)FINTIM
      WRITE(6,5004)FINTIM
 5004 FORMAT(7X,1PE10.3)
      IF(TIME.GT.FINTIM) GOTO 860
      START=1.
      READ(5,5005)DELT
 5005 FORMAT(5X,1PE8.1)
      WRITE(6,713) DELT
  713 FORMAT(1H0,'DELT HAS BEEN CHANGED TO ',1X,1PE11.4,1X,'SECONDS'///)
      PTIME=0.
      GOTO 1100
 6000 CONTINUE
C     MTIME=20E-06
      IF(TIME.GT.(FINTIM+DELT))GOTO 860
      JPTIME=TIME*1.E6
      IF(JPTIME.GE.JP)GOTO 881
      GOTO 700
  881 PTIME=1.1
```

```
  700 CONTINUE
      IF(LNSKIP.GT.0) GOTO 710
      WRITE(6,709)
  709 FORMAT(/T6    ,'I'        ,T   12   ,'TIME'   ,T   24   ,'      '  ,
     1T        34   ,'RPSE   '  ,T   45   ,'       ' ,T   55   ,'ACCEL'  ,
     2T        67   ,'      '   ,T   79   ,'RPST   ' ,T   88   ,'      '/)
      LNSKIP=LNSKIP+1
  710 CONTINUE
      IF(PTIME.LT.1.) GOTO 880
      WRITE(6,4000)    I      ,  TIME    ,  RPSE    ,  ACCEL   ,  RPST
      JP=JP+10
 4000 FORMAT(1X,I6,1PE13.6,T34,1PE13.6,T55,1PE13.6,T79,1PE13.6)
      PLNCNT=PLNCNT+1
      LNSKIP=LNSKIP+1
      IF(LNSKIP.EQ.101) LNSKIP=0
      PTIME=0.
  880 CONTINUE
      JGTIME=TIME*1.E6
      IF(JGTIME.GE.JG)GOTO 390
      GOTO 395
  390 GTIME=1.
  395 CONTINUE
      IF(GTIME.LT.1.) GOTO 385
      GTIME=0.
      LNCNT=LNCNT+1
      IF(LNCNT.GT.2001)GOTO 860
      LNC=LNCNT
      NCURVE=3
      P(LNC,1)=TIME
      P(LNC,2)=RPSE
      P(LNC,3)=ACCEL
      P(LNC,4)=RPST
      JG=JG+20
      DO 610 J=1,NCURVE
      IF(P(LNC,J+1)-MAX(1,J)) 620,620,630
  630 MAX(1,J)=P(LNC,J+1)
      MXTIME(1,J)=P(LNC,1)
  620 CONTINUE
      IF(P(LNC,J+1)-MIN(1,J)) 640,650,650
  640 MIN(1,J)=P(LNC,J+1)
      MNTIME(1,J)=P(LNC,1)
  650 CONTINUE
  610 CONTINUE
  385 CONTINUE
      IF( LNC.GT.1000) GOTO 860
   84 FORMAT(F15.3)
      IF(RPSEVR.LT.TEMRVR)GOTO 4997
      TEMRVR=RPSEVR
```

```
      RRTIME=TIME
 4997 CONTINUE
      IF(ACELVR.LT.TEMAVR)GOTO 998
      TEMAVR=ACELVR
      AATIME=TIME
  998 TIME=TIME+DELT
C*********************************************************************
  860 NPT=NCURVE*LNC
      WRITE(6,4000)I,RRTIME,ZERO,ZERO,TEMRVR
      WRITE(6,4000)I,AATIME,ZERO,ZERO,ZERO,ZERO,ZERO,ZERO,TEMAVR
      WRITE(6,605)
  605 FORMAT(1H1)
      WRITE(6,333) NPT
  333 FORMAT(1H0,T2,'NUMBER OF POINTS TO BE PLOTTED IS', I6//)
      WRITE(6,708) PLNCNT
  708 FORMAT(1H0,T2,'NUMBER OF LINES PRINTED IS',2X,F8.0//)
      WRITE(6,4996)
 4996 FORMAT(1H0)
      READ(5,4995)(TITLE1(K),K=1,60)
 4995 FORMAT(36A2/24A2)
      WRITE(6,4994)(TITLE1(K),K=1,60)
 4994 FORMAT(T3,3A2,T21,3A2,T37,3A2,T53,3A2,T69,3A2,T85,3A2,T101,3A2,
     * T116,3A2,T109,3A2,T122,3A2)
      WRITE(6,4996)
      DO 600 I=1,LNC
      NCUR=NCURVE+1
  600 WRITE(6, 690) (P(I,J),J=1,NCUR)
  690 FORMAT(1PE12.4,1P9E16.4)
C     WRITE(6,660)
  660 FORMAT(///T3,'VARIABLE',T15,'MAXIMUM',T27,'1ST MAX TIME'
     |   ,T51,'MIN',T63,'1ST MIN TIME')
      DO 670 I=1,NCURVE
C     WRITE(6,680) MAX(1,I),MXTIME(1,I),MIN(1,I),MNTIME(1,I)
  680 FORMAT(/10X,F11.4,3X,1PE12.4,10X,0PF11.4,3X,1PE12.4)
  670 CONTINUE
      READ(5,675) (TITLE(J),J=1,60)
  675 FORMAT(36A2/24A2)
      WRITE(6,4998)NUM
 4998 FORMAT(/1X,'NUM= ',F15.0/)
      CALL STPLOT(NCURVE,LNCNT,TITLE,NNN,P)
 1000 STOP
      END
C*********************************************************************
C     SUBROUTINE  STPLOT
C
C

      SUBROUTINE STPLOT(NCURVE,LNCNT,TITLE,NNN,P)
```

```
      COMMON MAX(1,10),MXTIME(1,10),MIN(1,10),MNTIME(1,10)
      DIMENSION LINE(101),RANGE(10),SCALE(10),OFFSET(10),P(NNN,4)
      REAL*4 MAX,MIN,MXTIME,MNTIME,LINE
      INTEGER*2 TITLE(60)
      REAL RL(10)/4HA    ,4HB    ,4HC    ,4HD    ,4HE    ,4HF    ,
     1            4HG    ,4HH    ,4HJ    ,4HI    /
      REAL DOT/4H.    /,BLANK/4H    /,PLUS/4H+    /
C
C   FIND RANGE OF EACH VARIABLE
C
      DO 1 J=1,NCURVE
C
C     THE NEXT LINE STOPS ANY UNDERFLOWS WHICH MIGHT BE GENERATED
C     BY SUBTRACTING MIN FROM MAX WHEN BOTH ARE EQUAL TO "0"
C
      IF(MIN(1,J).EQ.0.)GOTO 11
      RANGE(J)=1.*MAX(1,J)-1.*MIN(1,J)
      GOTO 12
C
C     FIND OFFSET TO MAKE MIN POINT PLOT AT BOTTOM OF GRAPH
   11 RANGE(J)=MAX(1,J)
   12 OFFSET(J)=MIN(1,J)
    1 CONTINUE
C
C     IF YOU WANT VARIABLE#1 TO BE PLOTTED ON A 1:1 BASIS INSERT
C     ON THE NEXT CARD (STARTING IN COLUMN 7)  "RANGE(1)=100."
      RANGE(1)=100.
      RANGE(2)=10.E4
      RANGE(3)=100.
C     (THIS IS EQUIVALENT TO A SCALE OF 10 UNITS/INCH PLOTTED ON THE
C     GRAPH)
C
C     CALCULATE GRAPHING SCALE FOR EACH VARIABLE IN UNITS/INCH
C
      DO 104 J=1,NCURVE
      IF(RANGE(J).LT.1.E-68) GOTO 105
      SCALE(J)=RANGE(J)/10.
      GOTO 104
  105 SCALE(J)=0.
  104 CONTINUE
      WRITE(6,103)
  103 FORMAT(1H1,///T 3,'IND VAR', T17,'DEP VAR',T33,'SYMBOL',
     * T47,'MAXIMUM',T64,'MAX TIME',T82,'MINIMUM', T101,'MIN TIME',
     *  T117,'SCALE(UNITS/INCH)'/)
      DO 102 I=1,NCURVE
      IP1=(I*6)-5
      IP2=IP1+5
  102 WRITE(6,110) (TITLE(K),K=IP1,IP2), RL(I),
```

```
     1 MAX(1,I),MXTIME(1,I),MIN(1,I),MNTIME(1,I),SCALE(I)
  110 FORMAT(T 3,3A2,T17,3A2,T35,A1,T45,1PE13.5,T62,1PE13.5,T80,1PE13.5,
     * T 99,1PE13.5,T116,1PE13.5/)
C
C  MAKE INITIAL LINE PRINTING
C
      WRITE(6,94)
   94 FORMAT(19X,'0....V....1....V....2....V....3....V....4....V....5...
     1.V....6....V....7....V....8....V....9....V....0')
      WRITE(6,93)
   93 FORMAT(19X,'+         +         +         +         +         +
     1 +         +         +         +         +')
      DO 200 LNC=1,LNCNT
      DO 91 I=1,101
   91 LINE(I)=BLANK
      DO 90 I=1,101,10
   90 LINE(I)=PLUS
      DO 89 J=1,NCURVE
      L=J+1
       SHIFT=P(LNC,L)-OFFSET(J)
      IF(RANGE(J).EQ.0.) GOTO 86
      K=100.*(SHIFT/RANGE(J))+1.5
      GOTO 85
   86 K=1
   85 CONTINUE
C
C     THE NEXT CARD PREVENTS OUTPUT FORMAT ERRORS FROM BEING GENERATED
C     IF AUTOSCALING HAS BEEN OVERRIDDEN AND THE VALUE OF K COMPUTED
C     FOR A VARIABLE IS GREATER THAN 101. IF SUCH IS THE CASE NO
C     CHARACTER WILL BE PLOTTED FOR THAT POINT.
C
      IF(K.GT.101)GOTO 89
      LINE(K)=RL(J)
   89 CONTINUE
      WRITE(6,88) P(LNC,1),LINE
   88 FORMAT(6X,1PE11.4,2X,101A1)
  200 CONTINUE
      RETURN
      END
SLASH STAR CARD NORMALLY GOES HERE.
//GO.FT06F001 DD SPACE=(CYL,(5,1))
//GO.SYSIN  DD *
100.
500000.
FINTIM=+5.150E-03
DELT=+2.0E-07
FINTIM=+5.150E-03
TIME  RPSE  ACCEL RPST
```

```
TIME  RPSE  TIME  ACCEL TIME  RPST                      * 00575
SLASH STAR CARD WOULD NORMALLY BE HERE.                 * 00576
                                                        * 00577
                                                        * 00578
```

BIBLIOGRAPHY

1. Bartee, T. C., Digital Computer Fundamentals, Second Edition, p. 76 - 81, McGraw-Hill Book Company, Inc., 1966.

2. Beckey, G. A. and Karplus, W. J., Hybrid Computation, p. 124, John Wiley and Sons, Inc., 1968.

3. Chu, Y., Digital Computer Design Fundamentals, p. 368 - 383, McGraw-Hill Book Company, Inc.,1962.

4. Doebelin, E. O., Measurement Systems: Application and Design, p. 274 - 281, McGraw-Hill Book Company, Inc., 1966.

5. Phister, M., Jr., Logical Design of Digital Computer, p. 128 - 129, John Wiley and Sons, Inc., 1967.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Documentation Center Cameron Station Alexandria, Virginia 22314	20
2.	Library Naval Postgraduate School Monterey, California 93940	2
3.	Professor James S. Demetry Department of Electrical Engineering Naval Postgraduate School Monterey, California 93940	2
4.	Professor George J. Thaler Department of Electrical Engineering Naval Postgraduate School Monterey, California 93940	1
5.	Mr. James H. Thach III "Topside" Mitchell Hill Road Old Lyme, Connecticut 06370	1
6.	Lt. John H. Bartol, Jr. 69 Dingletown Road Greenwich, Connecticut 06830	1

UNCLASSIFIED

Security Classification

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY *(Corporate author)* Naval Postgraduate School Monterey, California 93940	2a. REPORT SECURITY CLASSIFICATION Unclassified 2b. GROUP

3. REPORT TITLE

A Digital Tachometer/Accelerometer

4. DESCRIPTIVE NOTES *(Type of report and, inclusive dates)*

Master's Thesis, December 1969

5. AUTHOR(S) *(First name, middle initial, last name)*

John Hone Bartol, Jr.

6. REPORT DATE	7a. TOTAL NO. OF PAGES	7b. NO. OF REFS
December 1969	47	5

8a. CONTRACT OR GRANT NO. b. PROJECT NO. c. d.	9a. ORIGINATOR'S REPORT NUMBER(S) 9b. OTHER REPORT NO(S) *(Any other numbers that may be assigned this report)*

10. DISTRIBUTION STATEMENT

This document has been approved for public release and sale; its distribution is unlimited.

11. SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY Naval Postgraduate School Monterey, California 93940

13. ABSTRACT

Analog methods for measuring rotational shaft velocity and acceleration are well developed, yet present many drawbacks in practical use. A device is proposed to digitally compute estimates of the instantaneous velocity and acceleration of a shaft. The method of attack used in arriving at a final design proceeds from development of signal generating equipment through velocity- and acceleration-processing circuitry. Computer simulation is used to optimize the electronic circuitry to achieve the best accuracy.

DD FORM 1 NOV 65 1473 (PAGE 1)

S/N 0101-807-6811

Unclassified

Security Classification

A-31408

14 KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
ACCELEROMETER						
ACCELERATION MEASUREMENT						
ANGULAR ACCELERATION						
COMPUTER SIMULATION						
DIGITAL ACCELEROMETER						
DIGITAL TACHOMETER						
SPEED INDICATOR						
TACHOMETER						
VELOCITY MEASUREMENT						